

Fully Automatic Keyword Extraction with Text Classification by KNN considering Feature Similarity

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the full automation of the keyword extraction, using the text classification. In the previous work, it was required to present a category or a domain for performing the keyword extraction which is a domain dependent classification. The idea of this research is to classify a text which is given as the source of the keyword extraction into a domain and to extract keywords from it by the classifier which corresponds to the domain. The KNN which considers the feature similarities is adopted as the approach to both the text classification and the keyword extraction. The goal of this research is to automate the keyword extraction and improve the performance of both tasks, using the modernized KNN version.

I. INTRODUCTION

The keyword extraction is defined as the process of extracting important words from a text based on its contents. The keyword extraction is mapped into a binary classification which classifies a word into keyword or non-keyword. The process of keyword extraction is to index a text into a list of words, classify each word into keyword or non-keyword, and extract ones which are classified into keyword. The binary classification which is mapped from the keyword extraction belongs to domain dependent classification where a same word may be classified differently depending on a domain. It requires to present a domain as a tag for executing the keyword extraction.

This research is motivated by the necessity of solving the requirement of presenting the tag for executing the keyword extraction. The results from applying the proposed version of KNN algorithm to the keyword extraction was successful in the previous work. It is required to know the domain in advance for nominating the classifier which corresponds to it. This research is additionally motivated by the successful results from applying the proposed KNN version to the text classification. Connecting the keyword extraction with the text classification is the solution to the requirement of presenting the domain for executing the keyword extraction.

The idea of this research is to apply the KNN algorithm which considers the feature similarity to the connection of the keyword extraction with the text categorization. The KNN algorithm is modified into the version which is based

on the similarity metric between the numerical vectors which considers both the feature similarities and the feature value similarities. The modified KNN algorithm is applied to the text categorization for labeling a text with its own domain. A text is indexed into a list of words, the modified KNN algorithm is applied to the binary classification of each word into keyword or non-keyword, and words which are classified into keyword are extracted as the output. In this research, we do not need to present the domain for the keyword extraction system.

Let us mention the benefits from this research. The poor discrimination among the sparse numerical vectors is solved by using the feature similarities for computing the similarity between numerical vectors. The performance of both the text classification and the keyword extraction is improved by adopting the proposed version of KNN algorithm. It does not require to present the domain as a tag by connecting the keyword extraction with the text classification. The keyword extraction system is upgraded by automating the process of deciding the domain.

Let us mention some remaining tasks as the further research. We need to involve only some features in computing the feature similarities for reducing the complexity. Texts and words may be encoded into compound representations, each of which consists of more than one structured form. Other machine learning algorithms, such as Naïve Bayes and Support Vector Machine, may be modified into proposed version. The keyword extraction system may be implemented as a real program or a library by adopting the proposed approach.

II. PROPOSED APPROACH

A. Similarity Metric

This section is concerned with the process of encoding a word into a numerical vector. Texts are used as features for encoding a word so, and some are selected from a text collection. A similarity between texts is computed based on their shared words, and it is used as a similarity between features. The similarities among features are considered for computing the semantic similarity between words. In this section, we describe the process of encoding a word into a

numerical vector and the process of computing the similarity between words.

The process of encoding words into numerical vectors is illustrated in Figure 1. Texts are gathered as a collection from external sources as feature candidates. Only some texts are selected by a criterion among the gathered texts. In each word, its weights in the texts are computed as elements of the numerical vector. Refer to [2] for studying the process and the selection criteria.

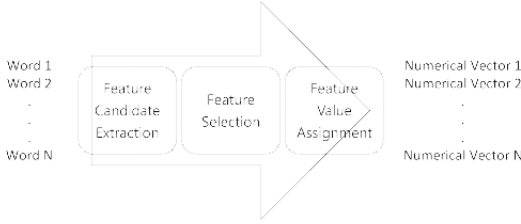


Figure 1. Process of Encoding Words into Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_d]$, $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_d]$, and $f_1, f_2, \dots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $\text{sim}(f_1, f_2)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as cosine similarity. The similarity between words is computed by considering the two kinds of similarities.

$$\begin{aligned} \mathbf{x} &= [x_1, x_2, \dots, x_d] \\ \mathbf{y} &= [y_1, y_2, \dots, y_d] \end{aligned} \quad \begin{array}{l} \text{Feature} \\ \text{Similarity} \\ \\ \text{Feature} \\ \text{Value} \\ \text{Similarity} \end{array}$$

Figure 2. Frame of Computing Similarity between Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_d]$, $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_d]$, and $f_1, f_2, \dots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $\text{sim}(f_1, f_2)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as cosine similarity. The similarity between words is computed by considering the two kinds of similarities.

Let us mention the process of computing the similarity between numerical vectors as the semantic similarity between words. The similarity between features, f_i and f_j , $\text{sim}(f_i, f_j)$, is abbreviated into s_{ij} . The similarity between

features is computed by equation (1),

$$s_{ij} = \frac{2 \times |D_i \cap D_j|}{|D_i| + |D_j|} \quad (1)$$

where D_i is the set of words in the feature, f_i , and D_j is the set of words in the feature, f_j , and the similarity matrix with the d features is constructed as a $d \times d$ square matrix, as shown in equation (2),

$$\mathbf{S} = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1d} \\ s_{21} & s_{22} & \dots & s_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & s_{dd} \end{pmatrix}. \quad (2)$$

The similarity between two numerical vectors, $\mathbf{x}$ and $\mathbf{y}$, is computed by equation (3),

$$\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^d \sum_{j=1}^d s_{ij} x_i y_j}{d \cdot \|\mathbf{x}\| \cdot \|\mathbf{y}\|}. \quad (3)$$

Equation (3) is used for computing the similarity between a novice word and a sample word in the proposed version of the KNN algorithm.

Let us make some remarks on the encoding process the process of encoding words into numerical vectors and computing the similarity between them. A word is encoded into a numerical vector by the feature candidate extraction, the feature selection, and the feature value assignment. The similarity between features which are given as texts is computed by equation (1). The similarity between numerical vectors which represent words is computed, considering the feature similarities by equation (3). In future, we will consider computing the similarities among some features rather than all features, for improving the computation speed.

B. Feature Similarity based KNN Algorithm

This section is concerned with the KNN algorithm which considers the feature similarities. The version was initially proposed as the approach to the text classification by Jo in 2019 [2]. The similarity metric between numerical vectors which was described in the previous section is used for computing the similarity between a novice vector and a training example. The labels of its nearest neighbors are voted with their identical weights for decoding the label of a novice vector. This section is intended to describe the proposed version of the KNN algorithm.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into numerical vectors, and they are notated by a set $Tr = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. A novice word is encoded into a numerical vector notated by $\mathbf{x}$. Its similarities with the numerical vectors which represent the sample words are computed by equation (3), as $\text{sim}(\mathbf{x}, \mathbf{x}_1), \text{sim}(\mathbf{x}, \mathbf{x}_2), \dots, \text{sim}(\mathbf{x}, \mathbf{x}_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

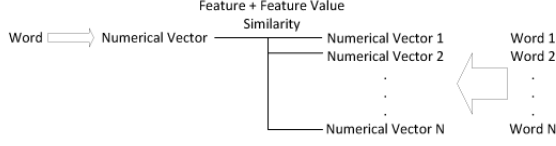


Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (4),

$$Nr = \text{Select}_k(Tr, \mathbf{x}), Nr \subseteq Tr \quad (4)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (5),

$$Nr = \{\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \dots, \mathbf{x}_k^{near}\} \quad (5)$$

The elements in the set, Nr , are used for voting their labels in the next step.

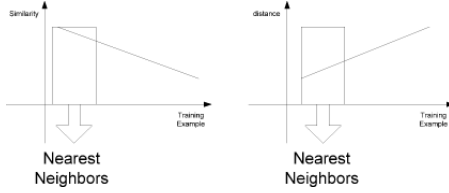


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(\mathbf{x}_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, $\mathbf{x}$, is decided by equation (6),

$$\text{label}(\mathbf{x}) = \underset{j=1}{\text{argmax}}^m \text{Count}(Nr, c_j) \quad (6)$$

The process of deciding the label of a novice item by equation (6), is called voting.

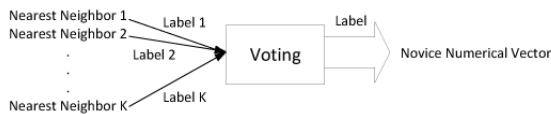


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the proposed version of KNN algorithm. It computes the similarities of a novice item

with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the keyword extraction system which adopts the KNN algorithm which considers the feature similarities. In Section II-B, we described the proposed version of KNN algorithm as the approach to the keyword extraction. It is viewed as a binary classification which classifies a word into keyword or non-keyword. This section is intended to describe the keyword extraction system with respect to its function and its architecture.

The process of collecting the sample words and encoding them into numerical vectors for implementing the keyword extraction system. The keyword extraction is viewed as a binary classification which classifies a word into keyword or non-keyword is illustrated in Figure 6. The topic-based word classification belongs to the domain independent classification where a same word is always classified identically with regardless of the domain, whereas the keyword extraction is the domain dependent classification where a same word may be classified differently depending on the domain. The words which are labeled with keyword or non-keyword are collected domain by domain. It is required to tag the input text with its own domain for executing the keyword extraction.

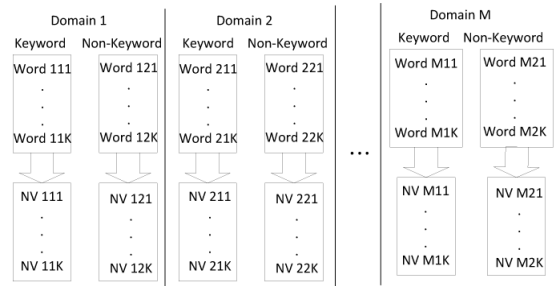


Figure 6. Preparation of Training Examples

The entire architecture of the proposed keyword extraction system is illustrated in Figure 7. A text is given as the input, and words are extracted from it in the indexing module. In the encoding module, sample words in the keyword group and the non-keyword group and ones which are indexed from the text are mapped into numerical vectors. The words which are indexed from the text are classified into one of the two categories in the similarity computation module and the voting module. The system consists of the four modules: indexing module, encoding module, similarity computation module, and voting module.

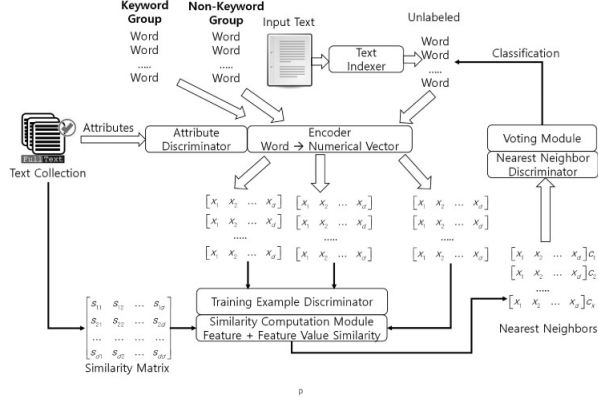


Figure 7. System Architecture

The execution process of the proposed system is illustrated as a block diagram in Figure 8. Sample words which are labeled with keyword or non-keyword are collected within a domain, and they are encoded into numerical vectors. An input text is indexed into a list of words, and they are also encoded into numerical vectors. The nearest neighbors are selected by the similarity computation, and the label is decided by voting ones of their nearest neighbors for each word. The words which are classified into keyword are extracted as the final output.

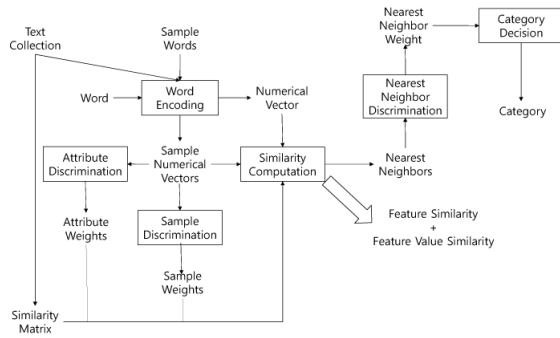


Figure 8. System Flow

Let us make some remarks on the proposed system which is illustrated in Figure 7 as its architecture. We propose the similarity metric which considers both the feature similarity and the feature value similarity. The poor discrimination among numerical vectors which is caused by their sparse distributions is solved. The classification performance is expected to be improved by what is proposed in this research. In the next research, we present the graphical user interface and the source code which are necessary for implementing the system as the complete version.

D. Connection with Text Classification

This section is concerned with the connection of the keyword extraction with the text classification. In the previous

section, we mentioned the keyword extraction system as an instance of domain dependent classification. The domain is automatically decided for nominating a classifier by connecting the keyword with the text classification. The KNN algorithm which was described in Section II-B is used for implementing the text classification. This section is intended to describe the connection of the keyword extraction system with the text classification for automating the decision of the domain.

The system architecture of the text categorization system which consists of the encoding module, the similarity computation module, and the voting module is illustrated in Figure 9. The encoding module is for encoding texts into numerical vectors by the process which is described in Section II-A. The similarity computation module is for computing the similarities between numerical vectors which represent a novice text and a sample text and selecting ones with their highest similarities as the nearest neighbors. The voting module is for deciding the label of the novice item by voting the labels of the nearest neighbors. This system is used for automating deciding the domain of the input text.

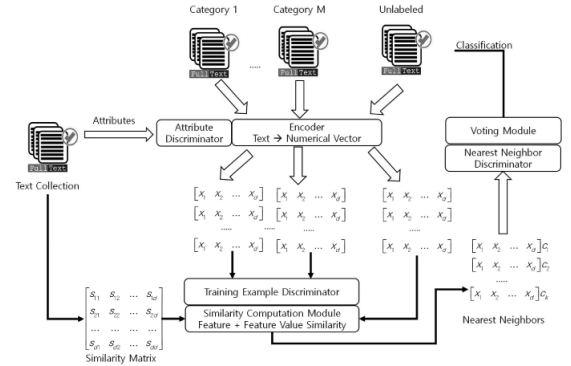


Figure 9. Text Categorization System

The connection of the keyword extraction with the text classification is illustrated in Figure 10. The texts each of which is labeled with its own domain are prepared as the sample texts, and the approach to the text classification is trained with them. A domain is assigned to a novice text by classifying it into a domain automatically. The novice text is provided with its domain to the keyword extraction system which is described in Section II-C. The role of the text classification system is to nominate the classifier which corresponds to the domain for the keyword extraction, automatically.

The process of executing the keyword extraction, connecting with the text classification is illustrated in Figure 11. Sample texts each of which is labeled with its own domain are prepared, and sample words each of which is labeled with keyword or non-keyword are prepared in each domain. A novice text is classified into one among the predefined domains, and the classifier which corresponds to the domain

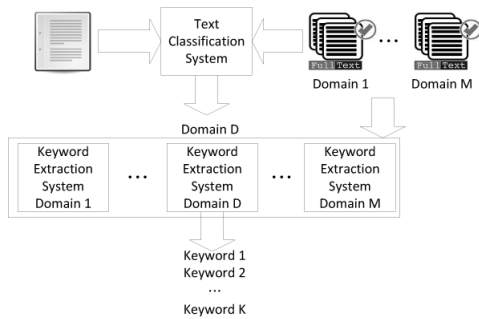


Figure 10. Keyword Extraction System connected with Text categorization

is nominated. The novice text is indexed into a list of words, and each word is classified into keyword or non-keyword. The list of words which is classified into keyword is the final output from this system; the domain which is assigned to the input text is the guide for selecting a classifier.

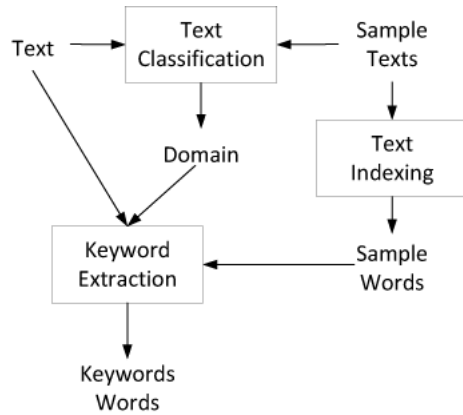


Figure 11. System Flow of Keyword Extraction connected with Text Categorization

Let us make some remarks on the connection of the keyword extraction with the text classification. The text classification is applied to the automatic decision of the input text domain. The role of text classification is to decide the domain for nominating a classifier, and the role of the keyword extraction is to extract important words from a text. In the execution flow, the input text is classified into a domain, it is indexed into a list of words, and each word is classified into keyword or non-keyword. We consider connecting the text classification as the main task the keyword extraction as the opposite to what is proposed in this research.

REFERENCES

- [1] T. Jo, "Extracting Keywords from News Articles using Feature Similarity based K Nearest Neighbor", 68-71, The Proceedings of International Conference on Information and Knowledge Engineering, 2018.

- [2] T. Jo, "Text Classification using Feature Similarity based K Nearest Neighbor", 13-21, AS Medical Science, 3, 4, 2019.